Procedia Computer Science

Volume 29, 2014, Pages 134–144

ICCS 2014. 14th International Conference on Computational Science

Performance-Aware Energy Saving Mechanism in Interconnection Networks for Parallel Systems

Hai Nguyen, Daniel Franco, and Emilio Luque

Computer Architecture & Operating Systems Department
Universitat Autònoma de Barcelona, 08193 Bellaterra, Barcelona, Spain
hai.nguyen@caos.uab.es, daniel.franco@uab.es, emilio.luque@uab.es

Abstract

The growing processing power of parallel computing systems require interconnection networks a higher level of complexity and higher performance, thus consuming more energy. Link components contributes a substantial proportion of the total energy consumption of the networks. Many researchers have proposed approaches to judiciously change the link speed as a function of traffic to save energy when the traffic is light. However, the link speed reduction incurs an increase in average packet latency, thus degrades network performance. This paper addresses that issue with a performance-aware energy saving mechanism. The simulation results show that the proposed mechanism outperforms the energy saving mechanisms in literature.

Keywords: energy saving; performance-aware; interconnection network; distributed systems

1 Introduction

Parallel and Distributed Computing Systems open the door to address open grand challenge problems in Science and Engineering. The ever growing amount of data to be processed demands a larger number of processing units; according to the statistics published by TOP500 on November 2013, the top supercomputer Tianhe-2 reaches a performance of 33.86 Petaflop/s with 3.120.000 cores [2]. The massive number of processing units puts interconnection network systems under the pressure to increase performance to better accommodate the communication among them. The higher level of network complexity and higher link bandwidth increase the energy consumed and the heat emitted. Empirical data show that a 10 degree increase in temperature results in a doubling rate of system failure, which reduces the realibility of the system. The energy cost and the heat dissipation problems in interconnection networks necessitate future network systems be built with much more efficient power than today. As a result, low power has become a priority design requirement for implementing scalable interconnected parallel systems [6].

Among many components, links contribute a substantial portion of the total power usage of an interconnection network. For instance, the link components in an IBM Infiniband 8-port

 Selection and peer-review under responsibility of the Scientific Programme Committee of ICCS 2014

doi:10.1016/j.procs.2014.05.013

switch with 12x links consumes 64% of the switch power [12]. The average link utilization in many communication systems - already quite low - tended to go down as the link speed increases [11]. For most traffic patterns, link utilization distributes non-uniformly over an interconnection network system, in both temporal and spatial terms. During operation, some phases require high link usage due to the high communication volume, whereas in other phases links almost idle. At a given time, links also separate into two groups: one group includes links that are in use, another group includes those are idle. In current commercial systems, a link consumes the amount of energy that is practically insensitive to the load it carries; it burns the same amount of energy even when it idles. Thus, link components consume energy proportionally to the traffic load become a desired behavior.

There have been many approaches to address the link power management problem. One popular energy saving proposal is the Dynamic Link Shutdown mechanism that turns off some underutilized links when the traffic load is light, and turns them on again when the traffic increases [15, 4, 8, 13, 16]. However, shutted-down links complicate the routing algorithm and connectivity due to the change in the network topology. Another proposal is the Dynamic Link Speed approach that judiciously adjusts the link speed according to the traffic load [10, 3, 14]. Links consume less energy when they are in low speed level. For example, the $1Gb/s$ Ethernet link consumes $1-2$ Watts, whereas the $10Gb/s$ Ethernet link exceeds 10 Watts [11]. Bit-serial technology is emerging where every link consists of several lanes, for example Infiniband specifies $4x$ and $12x$ link configuration, PCI-Express has 16 lane configuration denoted as $16x$. A physical link consists several lanes, which facilitates the link width and link speed adjustment, hence the Dynamic Link Speed approach comes naturally. This approach keeps the network topology unchanged, consequently simplifies the routing algorithm and the network connectivity issue. In this paper, we focus on the Dynamic Link Speed approach and extend it.

When the traffic load is low, the mechanism reduces the speed of links to save energy. A reduction in link speed leads to an increase in packet latency. Network performance derives directly from the average packet latency. The lower average packet latency means the higher performance of the network. Henceforth, in this paper we assess the network performance as a reversed metric of the average packet latency. We propose to boost the performance of the network by reducing the average packet latency when energy saving mechanisms are applied.

More specifically, this paper has following main contributions in introducing:

a) A simple way to avoid link flip-flop (link speed keeps alternating decrease with increase)

b) A two-level threshold policy to mitigate the latency penalty.

c) A Link Speed Aware Routing Policy to prioritize high-speed links & to boost performance.

The rest of the paper is organized as follows: Section 2 describes the Methodology, including the Energy Saving Mechanism in section 2.1 and the Performance Awareness in section 2.2. The hardware model implementation is depicted in section 3. Section 4 highlights the experimental evaluations. And the conclusion is drawn in section 5.

2 Methodology

The methodology in this paper includes two main parts. The *Energy Saving Mechanism* part describes the energy saving mechanism in literature extended by the flip-flop avoidance proposal. The *Performance Awareness* part introduces two proposals to boost the performance when applying the Energy Saving Mechanism. Fig. 1 shows the components which constitute the mechanism and the relationships among them. In this figure, the dashed boxes present the parameters that needs to be configured for the mechanism; the solid boxes present the components of the mechanism.

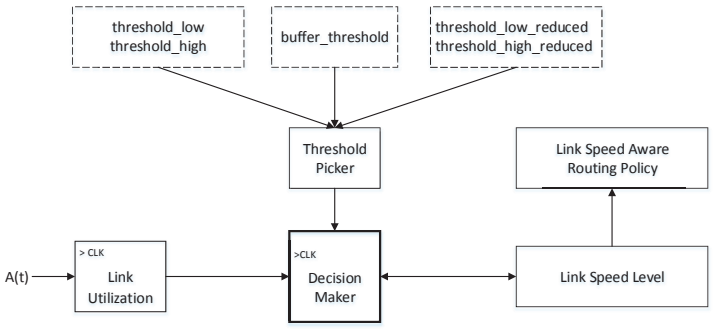


Figure 1: Performance Aware Energy Saving Mechanism

2.1 Energy Saving Mechanism

2.1.1 Monitoring and Decision Making

The energy saving mechanism consists two basic phases: The *Monitoring* phase and *Decision Making* phase. In the *Monitoring* phase, the mechanism monitors the Link Utilization (LU) at every link in both directions to independently prescribe whether to increase or decrease link speed. Equation 1 presents the mathematical description of LU for a link in a router.

$$LU = \frac{\sum_{t=1}^{H} A(t)}{H} \tag{1}$$

Where $\mathrm{A(t)} = \begin{cases} 1 \text{ if traffic passes the link in cycle } t \\ 0 \text{ if no traffic passes the link in cycle } t \end{cases}$

and H is a sliding history window size, in other words the mechanism only takes into account the link activities in the most recent H cycles..

LU directly measures how busy a link is. In the decision making process, the mechanism decreases the speed of a link if its LU drops below a threshold value of *threshold_low*; and increases the speed of the link if its LU exceeds a threshold value of *threshold_high*. There is a transition time with the value of *transition_time* link cycles for the change of link speed, thus the decision making process takes place every *decision_period* link cycles, which exceeds *transition_time* to allow the newly-speed-changed links to stabilize.

2.1.2 Link Flip-Flop Avoidance

According to the policy above in section 2.1.1, when the mechanism changes a link in speed level l_1 to a speed level l_2, if the traffic remains the same its LU changes $\frac{l_1}{l_2}$ times. For example, due to the low value of LU (below the value of *threshold_low*), a $2x$ link decreases its speed to $1x$ will increase its LU 2 times. The increase in LU might exceeds *threshold_high* and triggers the mechanism to increase the speed of the newly-speed-changed link back to $2x$ level. The increase in the link speed from $1x$ to $2x$ in turn halves the previous LU. This new value of LU drops below *threshold_low* as stated before, hence the mechanism will decrease the link speed in the next decision making phase. The same process happens again and thus causes flip-flop situation.

To avoid the link flip-flop problem, the mechanism triggers a link to change the speed not just by comparing the value of LU with the threshold values of $threshold\_low$ and $threshold\_high$, but it also needs to guarantee that the new value of LU falls between the values of the 2 thresholds. To achieve that behaviours, the mechanism imposes the following two rules:

1) A link decreases its speed level from l_1 to l_2 only if:

$$\begin{cases} LU < threshold\_low \\ LU * \frac{l_1}{l_2} < threshold\_high \end{cases} \Rightarrow \begin{cases} LU < threshold\_low \\ LU < \frac{l_2}{l_1} * threshold\_high \end{cases}$$
$$\Rightarrow LU < min(threshold\_low, \frac{l_2}{l_1} * threshold\_high)$$

2) A link increases its speed level from l_1 to l_2 only if:

$$\begin{cases} LU > threshold\_high \\ LU * \frac{l_1}{l_2} > threshold\_low \end{cases} \Rightarrow \begin{cases} LU > threshold\_high \\ LU > \frac{l_2}{l_1} * threshold\_low \end{cases}$$
$$\Rightarrow LU > max(threshold\_high, \frac{l_2}{l_1} * threshold\_low)$$

2.1.3 Aggressiveness and Responsiveness

The impact on the energy saving and network performance of the mechanism depends on the configured parameters. Parameters $threshold\_high$ and $threshold\_low$ directly impacts how the network performs. They form two characteristics for the mechanism:

- Aggressiveness: $aggressiveness = \frac{threshold\_high + threshold\_low}{2}$. This characteristic defines how aggressive the mechanism work to save energy. The higher the value of $aggressiveness$ the more easily the mechanism decrease the link speed, in turn the network achieves more energy saving with the expense of a possible increase in average packet latency.

- Responsiveness: $responsiveness = threshold\_high - threshold\_low$. This hysteresis band defines how responsive the mechanism reacts with the variance of the traffic. The higher the value of $responsiveness$ the more traffic variance required to trigger the mechanism to change the link speed.

The mechanism configures the value of $aggressiveness$ according to the objectives of the network. If networks provision applications that are latency insensitive then the $aggressiveness$ can be set at a high value to save more energy. Otherwise, $aggressiveness$ is tuned at a low value to prioritize network performance. The value of the $responsiveness$ should be configured to minimize the number of change in link speed while maintaining the dynamic of the mechanism in response to the traffic fluctuation.

2.2 Performance Awareness for Energy Saving Mechanism

An interconnection network system might apply the Energy Saving Mechanism described in section 2.1 to reduce the energy consumed by adjusting the link speed. In this system, average packet latency increases when packets move on lower bandwidth links, which deteriorate the network performance. In this section, we propose two methods to boost the performance (means to reduce the average packet latency) on top of the above-mentioned energy saving mechanism. Henceforth, we assume the network is coupled with the Dynamic Link Speed energy saving mechanism described in section 2.1 when introducing the two proposals below.

2.2.1 Two-level Threshold Policy

In high traffic situation, packets mostly spend time filling up the buffer space and waiting for other packets ahead of them in the buffer queue to move before making any progress. In this case, the additional latency can be hidden because the movement of packets are restricted by the availability of the buffer space, not the link speed itself; thus the impact of the saving mechanism on the packet latency is minimal. However, when the traffic is light, buffers are empty and the speed of packet movement depends on the speed of the links. Hence, any decrease in link speed results in an increase in the packet latency.

To mitigate this issue, we introduce a component called *Threshold Picker* in fig. 1 with a $two - level\ threshold$ policy. An additional pair of LU thresholds *threshold_high_reduced* and *threshold_low_reduced* feeds to the *Threshold Picker* component. The values of this pair are much lower than the original pair of thresholds. In light traffic situation, the link speed of a link decreases if only an extremely low traffic presents on it; the link remains in high speed even if just a small adequate traffic presents and thus reduces the latency penalty for the mechanism. In high traffic situation, the mechanism applies the original pair of thresholds to more aggressively save energy while negligibly impacts the average latency. Fig. 2 depicts the state changes of these two threshold values.

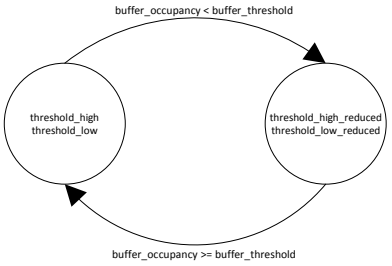


Figure 2: Two-level threshold policy

This policy needs to detect the traffic level presents on a link by relying on the buffer occupancy on the far end of that link. This information is freely available due to the widely-used credit-based flow control mechanism. This indicator acts as a litmus test for the mechanism to assess the traffic situation on a link and to prescribe whether to apply the reduced version of the LU thresholds. If the buffer occupancy drops below a $buffer\_threshold$ - means the traffic is extremely low - then the reduced version of the thresholds is applied, otherwise the original pair of thresholds takes place.

2.2.2 Link Speed Aware Routing Policy

We introduce a Link Speed Aware Routing Policy that prioritizes output ports coupled with links that are in high-speed level. This policy deploys on top of any reasonable adaptive routing algorithm and thus it is topology-and-routing-algorithm independent.

With the path redundancy in modern network infrastructure, a routing algorithm typically produces a set of several compatible virtual channels and output ports for a packet toward a destination. The packet will bid for a virtual channel from this set of virtual channels and output ports in the process of virtual channel allocation. The Virtual Channel Allocator, according to

its policy, will select a virtual channel in a port on which the packet will go through, taking into account the priority of the virtual channels and output ports.

At every routing step, the Link Speed Aware Routing Policy picks one output port from the pool of compatible ports $p_1, p_2, ..., p_k$. The ports couple with links corresponding to speed levels of $l_1, l_2, ..., l_k$ as described in fig. 3. The picked port will be given a higher priority compared with other ports in the Virtual Channel Allocation process and thus this port has a higher probability for the packet to go through.

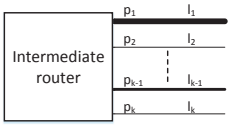


Figure 3: A pool of compatible output ports

The probability for a port p_i to be picked from a pool of compatible ports corresponds the speed level l_i of the link coupled with that port. The higher the value of l_i, the higher probability this routing policy will pick port p_i. Mathematically speaking, a port p_i has the probability of $\rho(p_i)$ as described in Equation 2

$$\rho(p_i) = \frac{l_i}{\sum_{p=1}^{k} l_p} \tag{2}$$

As a result, the ports coupled with links in high-speed levels have high priority in the virtual channel allocation process and thus contribute to reduce the average packet latency.

3 Hardware model implementation

Fig. 4 illustrates the hardware model implementation for the Performance-Aware Energy Saving mechanism. The Routing Unit introduces an additional *Link Speed Aware Routing Policy* component to prioritize high-speed links from a set of compatible links for a given packet as described in section 2.2.2. The *Dynamic Link Speed Adjustment* component sits between conventional router components and link components. It includes several extra supporting modules according to the above-mentioned proposals - The *Link Utilization* module monitors the link usage frequency. The link usage information is compared with the pair of threshold values (which are set according to the two-level threshold policy by the *Threshold Picker* module). The Decision Making process prescribes whether to adjust the link speed.

4 Experimental Evaluations

In this section, we evaluate the proposals in this paper by conducting a set of simulations on an extended version of the booksim simulator [1]. The interconnection network is configured with 64 processing nodes arranged in a 3D torus topology (4-ary 3-n torus) with virtual channel flow control. Packets are 8 flits in size. Every channel consists of 8 virtual channels. The values of *threshold_low* and *threshold_high* are configured with the values of 0.3 and 0.6, and their reduced version *threshold_high_reduced* and *threshold_low_reduced* are 0.1 and 0.2 respectively. The threshold for the buffer litmus test $buffer\_threshold$ is configured at the value of 0.05. The workload patterns and packet length depend on the type of applications.

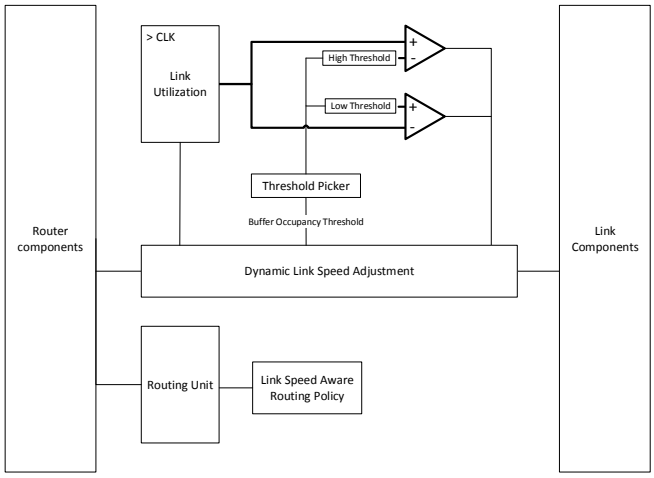


Figure 4: Hardware model implementation

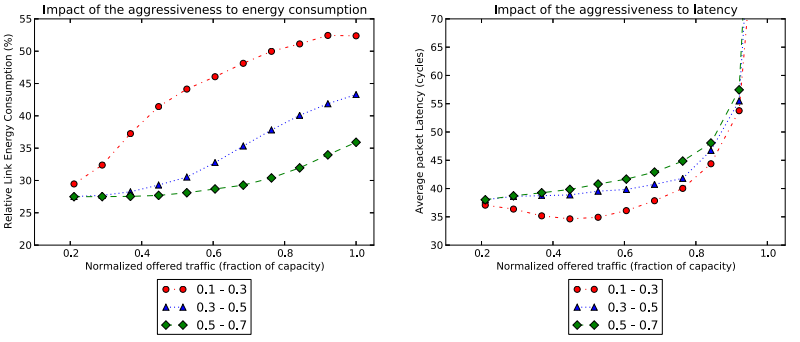
Two kinds of workload patterns were imported to the simulation system: synthetic traffic based on *transpose* distribution [7] and traffic from trace files of parallel applications. The Minimum Adaptive routing algorithm was deployed on this network simulation. These specific parameters and configurations are typical for an interconnection network and were selected for the purpose of illustration. Different configured parameters will yield different results, however the trend will be similar.

We assume that links consume energy proportionally with its speed level. The ratio between the energy consumed by link components when applying the energy saving mechanism over the energy consumed by link components when not applying the energy saving mechanism constitutes the Relative Link Energy Consumption of the network. We aim to minimize the Relative Link Energy Consumption and to maintain the network performance. In this section, we assume that the the network performance presents the reversed metric of the average packet latency - and thus we aim to reduce the average packet latency to boost network performance.

Experimental scenario 1: Evaluating the impact of the aggressiveness. Three different values of aggressiveness applying over an increasing range of offered traffic until the full network capacity. Fig. 5(a) highlights how aggressiveness impacts the Relative Link Energy Consumption. A higher value of aggressiveness (higher values of the threshold pair) results in a low relative link energy consumption, and vice versa. Fig. 5(b) reflects an opposite trend when higher aggressiveness leads to a higher value of average packet latency (and lower network performance). Thus, aggressiveness presents a typical tradeoff between the energy saving and network performance that an interconnection network must be configured. In our configuration, the aggressiveness corresponding to threshold values of $0.3 - 0.5$ shows the most balancing trade-off.

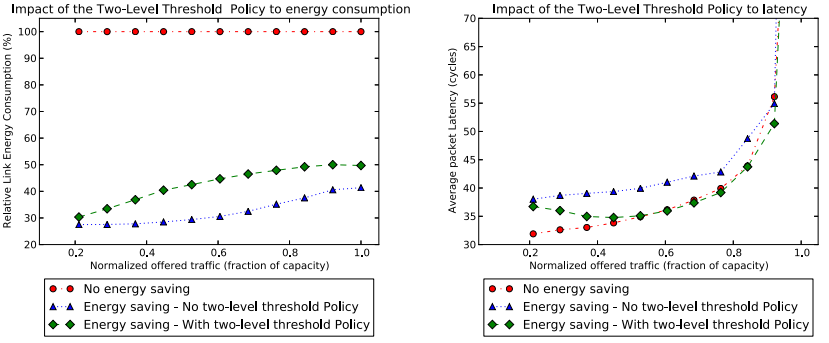
Experimental scenario 2: Evaluating the impact of the two-level threshold policy. In this scenario, we compare the behaviours of the default system without energy saving mechanisms, the system with the energy saving mechanism in literature and the system with the energy saving mechanism in literature coupled with the two-level threshold policy. As we can see, fig. 6 demonstrates the latency behavior and relative link energy consumption with an increasing range of traffic load. The two-level threshold policy significantly improves the average packet

latency behavior (and network performance) compared with the mechanism in literature.



(a) Relative Link Energy Consumption (b) Latency behavior

Figure 5: Evaluating the impact of aggressiveness



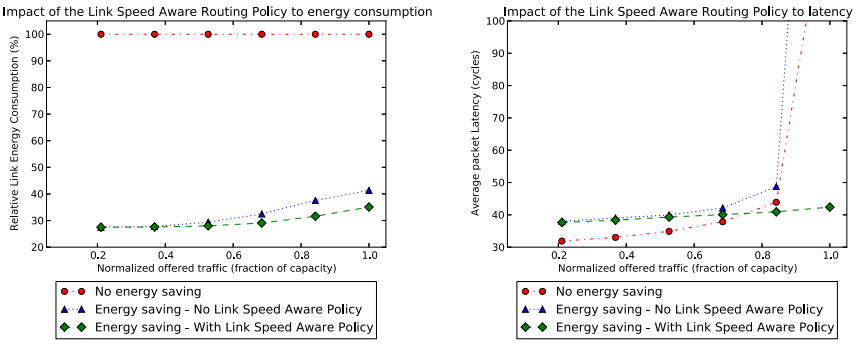
(a) Relative Link Energy Consumption (b) Latency behavior

Figure 6: Evaluating the impact of two-level threshold policy

Experimental scenario 3: Evaluating the impact of the Link Speed Aware Routing Policy. This scenario deploys the Link Speed Aware Routing Policy on top of an interconnection network system with the energy saving mechanism in literature. It then compares among the network systems with three configurations: no energy saving mechanism, in-literature energy saving mechanism, in-literature energy saving mechanism coupled with the proposed routing policy. Link Speed Aware Routing Policy shows an improvement in reducing the average packet latency (and thus improving the network performance), as shown in Fig. 7.

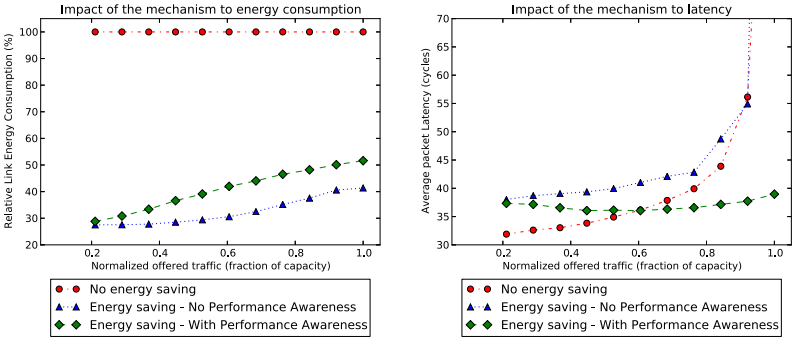
Experimential scenario 4: Evaluating the impact of the performance-aware energy saving mechanism. This scenario puts two proposals together and compare the energy saving and performance behaviors of the system without energy saving mechanism, the system with energy saving mechanism in literature and the system with our performance awareness deploying on top of an energy saving mechanism in literature. Fig. 8 shows that the network system better performs with our performance-aware energy saving mechanism.

We conducted another set of simulations to evaluate the impact of the energy saving mechanism with and without performance awareness on traffic imported from trace files. The traffic



(a) Relative Link Energy Consumption (b) Latency behavior

Figure 7: Evaluating the impact of the Link Speed Aware Routing Policy



(a) Relative Link Energy Consumption (b) Latency behavior

Figure 8: Evaluating the impact of the performance-aware energy saving mechanism

for the application Fluid Animate Particle Simulation using Smoothed Particle Hydrodynamics [5] was imported into the network with the same aforementioned network configuration using the Netrace framework [9]. Three set of simulations were carried out: the default network system, the network system with the energy saving mechanism in literature, the network system with the energy saving mechanism in literature coupled with our proposals in section 2.2. We put three simulation results under comparison.

As we can see from Table 1, the energy saving mechanism in literature reduces significantly the Relative Link Energy Consumption with a 19.42% increase in average packet latency. The performance-aware proposals in this paper help reduce the increased latency to 7.2% with a slight increase in energy consumption, thus outperforms the mechanism in literature. It reinforces our proposals in boosting the performance of the network by reducing the average packet latency.

Table 1: Impact of the Saving Mechanism & Performance Awareness (PA)

	No saving	**Saving without PA**	**Saving with PA**
Avg. Packet Latency	26.57 cycles	31.73 cycles	28.50 cycles
Avg. Latency Increase	0%	19.42%	7.2%
Energy Consumption	100%	25.03%	27.42%

5 Conclusion

In this paper we have introduced two proposals to improve the performance of the network when energy saving mechanisms are applied. Our proposals are self-sustained and are able to be deployed on top of any interconnection networks. Our future works focus on the impact of the proposals on different topologies and different traffic patterns.

Acknowledgments

This paper has been supported by the MEC-Spain under contract TIN2011-24384.

References

[1] Booksim interconnection network simulator, April 2013. `https://nocs.stanford.edu/cgi-bin/trac.cgi/wiki/Resources/BookSim`.

[2] Top500 supercomputer sites., November 2013. `http://www.top500.org`.

[3] D. Abts, M. R. Marty, P. M. Wells, P. Klausler, and H. Liu. Energy proportional datacenter networks. *SIGARCH Comput. Archit. News*, 38(3):338–347, June 2010.

[4] M. Alonso, S. Coll, J. Martinez, V. Santonja, P. Lopez, and J. Duato. Dynamic power saving in fat-tree interconnection networks using on/off links. In *International Parallel and Distributed Processing Symposium (IPDPS 2006)*, page 8 pp., april 2006.

[5] C. Bienia. *Benchmarking Modern Multiprocessors*. PhD thesis, Princeton University, January 2011.

[6] DARPA. Ubiquitous High Performance Computing (UHPC), March 2010. Broad Agency Announcement (BAA).

[7] J. Duato, S. Yalamanchili, and N. Lionel. *Interconnection Networks: An Engineering Approach*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002.

[8] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown. Elastictree: saving energy in data center networks. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, NSDI'10, pages 17–17, Berkeley, CA, USA, 2010. USENIX Association.

[9] J. Hestness, B. Grot, and S. W. Keckler. Netrace: dependency-driven trace-based network-on-chip simulation. In *Proceedings of the Third International Workshop on Network on Chip Architectures*, NoCArc '10, pages 31–36, New York, NY, USA, 2010. ACM.

[10] K. Kant. Power control of high speed network interconnects in data centers. In *Proceedings of the 28th IEEE international conference on Computer Communications Workshops*, INFOCOM'09, pages 145–150, Piscataway, NJ, USA, 2009. IEEE Press.

[11] K. Kant. Multi-state power management of communication links. In *Communication Systems and Networks (COMSNETS), 2011 Third International Conference on*, pages 1–10, 2011.

[12] J. Li, W. Huang, C. Lefurgy, L. Zhang, W. E. Denzel, R. R. Treumann, and K. Wang. Power shifting in thrifty interconnection network. In *Proceedings of the 2011 IEEE 17th International*

Symposium on High Performance Computer Architecture, HPCA '11, pages 156–167, Washington, DC, USA, 2011. IEEE Computer Society.

[13] A. G. Savva, T. Theocharides, and V. Soteriou. Intelligent on/off dynamic link management for on-chip networks. *JECE*, 2012:6:6–6:6, Jan. 2012.

[14] L. Shang, L.-S. Peh, and N. Jha. Dynamic voltage scaling with links for power optimization of interconnection networks. In *International Symposium on High-Performance Computer Architecture (HPCA-9 2003)*, pages 91 – 102, feb. 2003.

[15] V. Soteriou and L.-S. Peh. Design-space exploration of power-aware on/off interconnection networks. In *IEEE International Conference on Computer Design: VLSI in Computers and Processors (ICCD 2004)*, pages 510 – 517, oct. 2004.

[16] J. Yin, P. Zhou, A. Holey, S. S. Sapatnekar, and A. Zhai. Energy-efficient non-minimal path on-chip interconnection network for heterogeneous systems. In *Proceedings of the 2012 ACM/IEEE international symposium on Low power electronics and design*, ISLPED '12, pages 57–62, New York, NY, USA, 2012. ACM.